Typesetting in Bengali script using TEX

Palash B. Pal

Abstract

I will discuss various problems regarding typesetting in Indian languages, with special reference to the Bangla script. These problems include: conjunct vowel symbols; conjunct consonants; nonlinear placement of glyphs; minimum glyph set not unanimously agreed upon. I will also discuss how an almost phonetic transcription in the input file can produce true Bangla output.

1 Introduction

TEX was originally designed [1] to be an efficient typesetter for scientific documents. The basic, or the default, font was Roman, and mathematical symbols appeared as control sequences. Some diacritical marks and a few special symbols were also included in the original TEX, making it possible to typeset documents in most major languages of Western Europe. The macro packages derived from TEX, like the versatile LATEX or more specialized packages to meet the needs for a specific society or a specific journal, also inherit these characteristics of TEX. Henceforth, whenever I mention TEX, it will mean the original, unadorned TEX in addition with macro packages like LATEX.

The design of TEX fonts is done through METAFONT [2]. Once one masters this language, one can create any font containing any design, or shape, of "letters". Therefore, it was straightforward to create Cyrillic fonts, Gothic fonts etc., as well as many different font-shapes for the Roman font, such as the blackboard bold fonts, the calligraphic fonts, etc.

Apart from the easy typesetting of mathematical formulas, TEX has many useful features. I need not go through all of them, only indicate a few. TEX files do not have any hidden commands (what you see is what you have ordered); one can use macros to make typesetting simple; the contents, the running page headings etc can be programmed, and so on.

For many of these reasons, it was desirable to create fonts so that the languages using other scripts can also take advantage of TEX — not only to write mathematical articles in these scripts, but also for simple typesetting of the plain text of a novel, for example. In addition, several macro packages were necessary so that the default fonts, styles, etc., for TEX can be set for these scripts.

অ	আ	ই	ঈ	
উ	ঊ	ঋ		
এ	ঐ	ও	ঔ	
ক	খ	গ	ঘ	ঙ
চ	ছ	জ	ঝ	ঞ
ট	ঠ	ড	ঢ	ণ
ত	থ	দ	ধ	ন
প	ফ	ব	ভ	ম
য	র	ল	ব	শ
ষ	স	হ	ড়	ঢ়
য়	ৎ	ং	ঃ	ঁ

Figure 1: The Bangla alphabet. The vowels first, the consonants second.

Recently, I have developed such a package for our language Bangla* [3]. The purpose of this article is to outline the development of this package, and to discuss some issues arising from it.

2 The writing system

2.1 The alphabet

The Bangla alphabet derives from the Brahmi and Kharoshthi alphabets that were in vogue more than two thousand years ago. (The same can be said about most other alphabets used in India today.) The arrangement of letters in these alphabets follow a very logical pattern — the vowels and the consonants are kept separately, and the consonants are arranged according to the point of articulation of the corresponding phonemes. In Fig. 1, I show the Bangla alphabet.

To create the fonts, one might start with coding the letters in the alphabet, but troubles arrive even at this stage. Some people might want to say that in the list of vowels, there should be one more, indicated by the suspicious gap in the second row of vowels. Indeed, in older primers of the language, this space used to be occupied by another "letter" which stood for the vocalic /l/ sound as in the final syllable of the English word *little*. This was a legacy of the Vedic language, and neither the letter nor the sound was ever used in Bangla. Even those who would like to include it in the alphabet would agree that it has no function in writing Bangla. So most primers and dictionaries now omit this letter from the alphabet, and we find it reasonable to do so.

Once this decision is taken, one can design the letters in a straightforward manner. However, the story just starts there. There are two important issues to be addressed before the font design can be complete. First, we see that the number of vowels and consonants are much larger than that in the Roman alphabet. So one has to decide how to access the letters from the keyboard. Second, a font does not contain only letters. There are the punctuation marks, the numerical symbols, possibly some diacritical marks, and so on. All such symbols, together with the letters, are called *glyphs*. Before constructing a font for a certain script, one thus has to determine what is the minimum glyph set. This is a non-trivial task for an Indian script like Bangla, as we will discuss in detail below. Most of these issues are not specific to Bangla, but are common to other Indian scripts.

We will address the second issue first, and then come back to the first one.

2.2 Conjunct vowel symbols

We have already shown the vowels in Fig. 1. However, these symbols are used only if the vowel sound occurs alone in a syllable, or at the beginning of a syllable, i.e., if the syllables are of the form [V] or [VC].† Examples:

আম	*ām*	(mango)
উট	*uṭ*	(camel)

However, if the syllable has a vowel elsewhere, e.g., has the form [CV] or [CVC], the vowel symbol attaches itself to the preceding consonant symbol, and takes a different form. For example:

মা	*mā*	(mother)
টুপ	*ṭup*	(sound of dripping water)

Notice the differences. The /ā/ sound is written as 'আ' in a [V] or [VC] syllable, but only as 'া' attached to the preceding consonant symbol in [CV] or [CVC] syllables. The sound /u/ appears as a little attachment to the bottom of a consonant if this consonant is in the same syllable. And so on for other vowels as well.

So far, this means that we need to double the number of vowel glyphs to accommodate the conjunct vowel symbols as well as the pure vowel symbols. It need not be as simple as that, as we will see.

* Bangla (or "Bengali", as it is often written in English) is a language with more than 200 million native speakers. It is the national language of Bangladesh, and the state language of two Eastern states of India — West Bengal and Tripura. The Bangla script, with minor variations on a few letters, is used to write other languages of northeastern India, e.g., Assamese.

† We will denote syllables by square brackets, where V will denote a generic vowel sound and C a consonant sound.

2.3 Conjunct consonant symbols

The consonants also often do not appear in their simplest form in writing. If two consonant sounds occur within a word which are not separated by any vocalic sound, the two are often represented by a conjunct symbol. Let me provide a few examples:

ন+দ = ন্দ, প+প = প্প, স+ম = স্ম

Roughly speaking, the first consonant takes an abbreviated form which attaches itself to the top or to the left of the second consonant symbol.

Although not all possible combinations of two consonants are used in writing, the number of allowed combinations is huge. Moreover, there can be conjuncts of three consonants as well, although in this case the third one must be the member of a few select consonants: /y/, /r/, /l/, /w/. For example:

স+ত+র = স্ত্র, ত+ত+ব = ত্ত্ব, ন+দ+র = ন্দ্র

All such combinations need to be accommodated in the font in some way or other, and it is no easy task.

2.4 A short history of the Bangla glyph set

Large scale printing started in Bengal in the 19th century. From the beginning to about the middle of the 20th century, printing was done exclusively on letter presses. In these machines, one used a rectangular matrix mold for each character. These matrices were arranged one after another to produce a line of text. The matrix molds for one line of text were then held between two metal plates, usually called "lead". Such lines were then arranged vertically to form a page. All lines in a page were finally held together by a piece of string and the impression of the page was taken on paper.‡

The matrix molds for single characters could not be placed vertically on one another without a lead separating them. This posed a problem. Conjunct consonants are stacked vertically in most cases. It was impossible to use the matrices for single consonants and stack them vertically to obtain the conjunct consonants. Therefore, one had to have matrices not only for the simple consonants, but separate ones for each possible conjunct.

This list included not only the conjunct consonants, but some conjunct vowels as well. For example, consider the symbol for the sound /u/ in the non-initial positions in a syllable, which will be represented by the symbol /+u/. The symbol is called *u-kar*. It looks like a little knot below the consonant. It was impossible to produce this effect by using a separate matrix for the *u-kar*. One had to have separate matrices for /ku/, /bu/, /mu/, and many more. What was worse, one needed separate matrix molds for the *u-kar* attached to conjunct consonants such as /s+t/, /s+t+r/ and so on. The same problem existed for other conjunct vowels, and in each case one had to have a complete set of matrices for all consonants attached to this vowel. In the middle part of the 19th century when Ishwar Chandra Vidyasagar standardized the glyph set, about 1000 glyphs were necessary for typesetting Bangla.

The situation improved a bit with kerned molds, but the real revolution in Bangla printing occurred in the 1930s, when lino printing was introduced. In a lino machine, one uses mirror images of the matrices for different glyphs, arrange them to form a line of text, and pour molten metal on it to produce a matrix mold for the entire line. The problem was that lino machines had room for only about 200 glyphs. Hence it was necessary to devise new strategies for typesetting Bangla in lino machines.

Two main strategies were employed. First, symbols like *u-kar* appeared not vertically to the bottom of a consonant but slightly to the bottom right corner of it. This eliminated a few dozens of matrix molds for *u-kar* attached to various consonants and conjunct consonants. If you add the reduction from other conjunct vowel symbols, the reduction amounted to several hundreds.

Second, the conjunct consonants themselves were typeset a little differently. Rather than the two consonant symbols joining in the vertical direction, the first consonant was typeset in a somewhat midget form to the left of the second one. For example,

ন+ত : ন্ত → ন্ত , স+ক : স্ক → স্ক

The advantage was that, the same glyph for a "broken" /n/ could now be used in front of any other consonant. So, instead of all glyphs of the form /n/ plus some other consonant(s), one could make do with just a single glyph for /n/.

With these two strategies, the number of necessary glyphs were reduced to the number that a lino machine could support, and Bangla typesetting entered into a new era.

2.5 Non-linear arrangement of glyphs

There was another big problem, which was not solved by lino printing.

Consider the conjunct vowel symbol for the vowel sounds /+e/ and /+i/. We show how they

‡ In fact, one did not take an impression of a single page but of several pages together, but this detail is irrelevant for us.

appear when they attach themselves to the consonant ক, representing the sound /k/:

ক+এ = কে, ক+ই = কি

The problem now shows. The conjunct vowel symbols for these two vowels appear *before* the consonants, although phonetically the vowel sounds appear later. In other words, the glyphs cannot be typeset in the order in which the sounds appear in the spoken word.

The problem continues with some other symbols for conjunct vowels. What happens with the /+o/ sound is curious:[§]

ক+ও = কো

Notice that the symbol for /+o/ appears in two parts — one part (which looks like the symbol for /+e/) goes to the left of the consonant, whereas the other (which looks like the symbol for /+ā/) goes to the right. There are two special conjunct symbols for the diphthongs /+oi/ and /+ou/, which also have this problem of non-linearity.

3 Matters of outlook

When one starts to design a typeface for any script, one has to face several questions; and if the typeface is to be constructed with a new technology, the questions become more intriguing.

Take an example from the Roman typeface. Typewriters had monospaced fonts, i.e., all characters had equal width. Consequently, the symbols for the number zero (0) and the capital letter O looked pretty similar, sometimes identical. This created a problem for computers, because substituting one for the other would send incorrect signals to the computer. However, looking at the file, it would be difficult to spot the error. So, in the early days of line printers, the number zero started to appear with a cross mark through it. This was a welcome change, because it reduced confusion.

Then came laser printers. Try to imagine a person trying to develop a font for this new technology. The print quality of laser printers is much better than that of typewriters or line printers, and the fonts need not be mono-spaced. So maybe, even without a cross mark through zero, the two symbols could be distinguishable. The font-maker would now face the question: should he go back to the traditional design where the difference would now be appreciable, or should he maintain the cross mark running through zero in order to make the distinction more pronounced, something that will be obvious even in handwriting? To my knowledge, all laser fonts have taken the former option, and I think it is a pity. There are similar issues regarding '2' and 'z' (where the second one is often hand-written with a horizontal cross mark in the middle in order to avoid confusion), or regarding the number one (1) and the lowercase letter 'ell' (l), which had a single key on most typewriters.

There will be many such questions for Bangla. One class of questions would involve the shapes of the glyphs. As I already mentioned, lino printing simplified the shapes a lot. The change in that case was prompted by the restriction on the total number of glyphs. In a computer font, one can accommodate much larger number of glyphs. Therefore the first question is: should we forget about the reforms implemented by lino printing, or should we go back to the traditional designs?

The answer need not be a clear yes or no. In some cases, it is easy and convenient to go back to the traditional design. Consider for example the issue about the *u-kar*. It is easy for METAFONT to define a glyph with zero width where the character appears to the left of the cursor. In other words, while processing the *u-kar*, the cursor does not move further to the right, so that the symbol for *u-kar* appears below the preceding consonant.

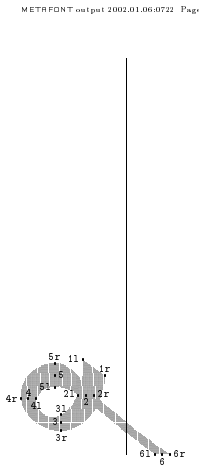

Figure 2: Design for the conjunct vowel symbol for /+u/. The vertical line denotes the cursor.

In some other cases, however, I felt it would be preferable to keep the lino reform. There were some conjunct consonants which, in the traditional method, were *non-transparent*. This means that, looking at the glyph for the conjunct consonant, one could not make out which consonant sounds they contain, because the glyph for the conjunct did not

[§] By /o/, we represent the vowel sound of the word 'role', not that of 'rock'.

resemble a superposition of the glyphs of the component consonants. In the lino style, such glyphs were discarded in favor of two different glyphs, a "broken" one for the initial consonant and the full one for the later one. They were therefore *transparent*, and it would make more sense to keep these reformed shapes.

In fact, in some cases we can do better than the lino. Consider for example the symbol for the sound /r/ when it appears as the final component of a conjunct consonant cluster (we will call it the symbol for /+r/). It then shows as a curvy line at the bottom of the rest of the cluster, like this:

প+র = প্র, গ+র = গ্র

Lino fonts had to have separate glyphs for each such conjunct. However, in our computer fonts, only one curvy line, extended to the left of the cursor in a zero-width character, can do most of the job. Notice that consonants like প (/p/) or গ (/g/) have a vertical bar near the right end. If the distance between this vertical bar and the right end is kept the same for all such consonants, it is trivial to adjust the glyph for /+r/ so that it attaches exactly to the bar.

An important comment has to be made here. It is to be understood that we are trying to reduce the number of glyphs, but *not* in order to avoid more work in the design stage. Neither are we doing it as a desperate solution for accommodating all glyphs into a standard font which can contain 256 characters. The point is that one doesn't need unnecessary complication. A more advanced technology should be capable of making more adjustments, and so a fewer number of glyphs might be able to produce the same (or even a better) output than what could have been done by a less advanced technology with a larger number of glyphs.

4 Input of characters

4.1 Posing the question

We now come to a question which was raised earlier but was not addressed. After we create the font, how do we input the characters into a file?

The question is serious for Bangla and other Indian languages firstly for the sheer number of glyphs. Even after cutting down on the number through various techniques as described above, it is not possible to fit all glyphs on a keyboard unless one is ready to deal with a much larger keyboard. This is the main reason why good typewriters did not develop for the Indian scripts. As a result, the input procedure of characters remains unsettled.

However, the number of glyphs is not the only problem. There are others. Take, for example, the letters corresponding to the sibilant sounds. In the Roman alphabet, there are two of these: /s/ and /z/ — the first one unvoiced and the second voiced. In Bangla, there is no voiced sibilant consonant. There are two unvoiced sibilant phonemes, a palatal sound (like /sh/ in English 'shot') and an alveolar one (like /s/ in 'sit'), and two letters for them. However, Sanskrit had also a retroflex sibilant sound and a letter for it, as do some modern South-Indian languages. Bangla and other northern Indian languages do not really have the retroflex sound but carry the letter as a legacy to Sanskrit. So, in total, there are three sibilant consonants. Add with them the 'broken' forms which can be used as the initial component in a conjunct consonant, and you have six. Let us now ask the simple question: how are we going to input these six different things into a computer file?

This is less of a problem in WISYWIG word processors. You could have the six things bound to certain keys so that pressing those keys would give you these characters on the screen. But for TEX, the issue is different. Here, the input file contains ASCII characters only. Thus it is desirable to have the input file in such a way that the phonetic correspondences are recognizable. In other words, if we choose the upper-case 'Y' to represent the letter for the alveolar /s/, it would not be very nice because the source file will not be readable. It would be extremely difficult to edit such a file.

4.2 Use of control sequences

To my knowledge, there are two alternatives for producing a TEX input file which is at least roughly phonetic. One is to use control sequences.

This was indeed the strategy taken by an earlier Bangla font designed by Abhijit Das [4]. To use his font, one can type `\kh` to obtain the consonant 'খ', or `\th` to obtain 'থ'. Indeed, the consonant 'খ' pronounces like an aspirated /k/, and 'h' can be taken as the indication for aspiration. The usual rules for the Roman transcription of Bangla (and other Indian) names uses 'kh' to denote this letter.

So this looks like a pleasant solution, until you realize that TEX has to know where the control sequence ends, so you need to leave a space in the input file right after using this sequence. The space has to appear even if the intended letter appears at the beginning or in the middle of a word. Consequently, the words will not appear together in the input file, and it will be a constant problem while trying to edit the file. If, on the other hand,

one protects the control sequence within braces, it will mean too much typing for producing just one letter — five to be precise, including the open and closed braces, the backslash, and the letters 'kh'.

The use of control sequences terminating with a non-letter character could have been a more economic solution. For example, the macro for the intended character could have been `\kh.` instead of `\kh` without the period. While typing, four keystrokes would have been needed, and words could have been kept together as well. But this solution was not taken by the package created by Das [4].

Of course it is not mandatory to input the letter as a control sequence; instead, one may input the letter directly. In this case, the letter in question occupies position number 75 in his font, the number which is taken by the capital letter 'K' in ASCII. Thus, within the font, if one types 'K' (i.e., shift+k), one will also obtain the same letter. One can then take capitalization (in the input file) as a sign of aspirated consonants.

Unfortunately, this approach would not take us very far. In fact, the previously cited `\th`, which is an aspirated form of the consonantal soft /t/ sound and pronounces almost like the consonant in the English word 'thaw', occupies position number 122, which in ASCII is the place for lower case 'z'. If we have to write 'z' in place of the sound /th/ in the input file, we will be far from a phonetic input file. Moreover, since the sheer number of characters exceeds the number of keys on the keyboard, there will always be some characters which could not be typed directly from the keyboard.

4.3 Use of ligatures

For the reasons stated above, I have not used control characters at all. Instead, I have used ligatures. In the input file, if we have 'k', TEX gets ready to typeset the letter for the /k/ sound (क), but also receives information from the ligature table not to do it quite yet. If the next input character happens to be 'h', it forgets about the letter it picked up, and picks up the letter for the /kh/ sound (ख) instead.

The method is used for the Roman fonts as well, and is described in the METAFONT book [2]. However, the number of ligatures necessary in a Roman font is very few — 'fi', 'ff', 'ffi', 'fl' and 'ffl'. On the other hand, for my Bangla font, there are ligatures starting with about 80 characters, and in most cases there are multiple possibilities for the completion of the ligature.

Of course, when there are so many ligatures, one has to be very careful not to create *too* many. For example, if you want to use 'kh' as a ligature, you have to be careful that the need for typesetting 'k' and 'h' separately will either not arise at all or will arise only very rarely, in which case the ligature must be broken manually, by enclosing one of the letters within a pair of braces. This, in my opinion, was the hardest part in the design of the fonts, but so far I have not heard any complaint about it. For the most part, I have been able to use the standard combinations which are used in Romanized transcription of Bangla words.

Things would have been easier if METAFONT had one additional feature. In the ligature file, you can replace any combination AB (of course the letters are symbolic here) by only A, or by only B (i.e., tell TEX to ignore one of the components of a ligature), or by C (i.e., tell TEX to forget both and use a third one). However, I found no way of replacing AB by CD, i.e., by two (or possibly more) new characters. It could have helped very much, although I managed somehow without it.

4.4 The missing vowel symbol

There is another interesting aspect of the Indian scripts that we have not touched before. The first vowel of the alphabet, whose sound will be denote by /a/ (in Bangla, it sounds roughly like the vowel sound in the English word 'rock'), is left out of the written representation when attached after a consonant in a syllable. In other words, just the consonant letter written by itself implies that the vowel sound /a/ is attached to it. Thus, for example, if you want to typeset 'rock' in Bangla (there is such a word, although it means something completely different), it would be sufficient to typeset '`rk`'. But obviously that will not look very phonetic.

I found a solution to this problem when I realized that no special sign needs to be created for /+o/. As already mentioned, the symbol for /+o/ is a combination of the symbols /+e/ and /+ā/. Thus, the space for 'o' is "free" in some sense, and so I have put a null letter in its place. It means that if you type an 'o' in your source file, it will not produce anything in the output. Therefore, you can type '`rok`' to obtain the word which sounds like 'rok', and certainly it looks phonetic.

I have to make a special comment about other Indian languages here. In most other major languages of India, the first vowel is not pronounced as it is in Bangla. Rather, it is pronounced like a short /ā/, something like the vowel sound of the English word 'mud'. For these languages therefore, my solution will not be very attractive. For Bangla, however, it seems quite acceptable.

4.5 The question of non-linearity

Finally, the question of non-linear arrangements of characters has to be addressed. As I mentioned earlier, the conjunct vowel signs for /e/ and /i/ appear before the related consonant, and the sign for conjunct /o/ appears on both sides. For example, suppose we want to write the word 'pen' in Bangla. (This word has been borrowed from English, so it is a Bangla word as well.) How would we have to typeset it? Not as `pen`, but as `epn`. Needless to say, this is not phonetic.

I have used a control sequence to get rid of this trouble. I have made the following definition:

```
\def\*#1*#2{o\null{#2}{#1}}
```

In short, it reverses the order of two things. Thus, the word 'pen' can now be typeset as '`\*p*en`', because the symbol after the second star sign would be typeset first, as needs to happen for Bangla. The advantage is that one can disregard the star signs while reading from the screen, thus obtaining a phonetic transcription.

5 Afterthoughts

The quality of a font is of course ultimately determined by its users. It is not easy to foresee a time when commercial editions of Bangla novels would be typeset in TEX, just as it is difficult to find many English novels which have been typeset in TEX. However, there are many academic books in English and other European languages which are typeset in TEX. If this can be done in Bangla as well, it will be reap great benefits. The publishers will love it if the authors write their books in LATEX which they won't have to typeset. In a limited market like that for Bangla books, the resulting savings in the typesetting cost can prove to be very valuable.

The first Bangla book that came out in my fonts is a book on Bangla phonetics authored by me [5]. It is easy to say that it is the first, because it came out before I made the fonts public. It might even be the first book in any Indian language typeset in LATEX. I am happy to say that since then, a few other books have come out as well, including some which are not written by me. Let's see what lies in the future.

References

[1] D. E. Knuth, *The TEXbook*, Addison-Wesley Publishing Company, 1984.

[2] D. E. Knuth, *The METAFONTbook*, Addison-Wesley Publishing Company, 7th printing, 1992.

[3] Palash B. Pal, *Bangtex: A package for typesetting documents in Bangla using the TEX/LATEXsystems*, available primarily from `http://tnp.saha.ernet.in/~pbpal/bangtex/bangtex.html`.

[4] Abhijit Das, *Bengali Writer TEX Interface*, available from `http://www2.csa.iisc.ernet.in/~abhij/bwti/`.

[5] Palash B. Pal, ধ্বনিমালা বর্ণমালা (Sounds and Letters), Papyrus, 2001.

◇ Palash B. Pal
Theory Group, Saha Institute of Nuclear Physics
1/AF Bidhan-Nagar, Calcutta 700064
India
`pbpal@theory.saha.ernet.in`